MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

Department of Information Technology

Subject: Micro Processor Lab **Class: SE-IT**

Experiment No: 04 **Semester: IV**

Roll No: 466

---

1. Program to move set of numbers from one memory block to another.

Theory:

The value of counter which tells the number of bytes to be transferred is stored at offset 500. The 8-bit data which have to be transfer is stored in continuous memory location starting from 501. The data is transferred to a continuous memory location starting from 600. The value of DS and ES is taken equal to 0000. The program starts from offset 400.
CLD instruction is used to clear the directional flag, i.e., DF=0. Now, value of SI and DI will be increased

Instructions to be explained:

- MOV AX, @DATA – it is the first line of code that gets run. @DATA is a variable that holds the value of the location in memory where the data segment.
- MOV is used to transfer the data from memory to accumulator
- ADD is used to add accumulator with any of register
- DAA is used to check if sum > 9
- JNC is used jump if no carry to given memory location
- INR is used to increase given register by 1
- INT generates a software interrupt. It takes the interrupt number formatted as a

byte value.

- AND instruction is used for supporting logical expressions by performing bitwise AND operation
- ROL rotates the bits within the destination operand to the left, where left is toward the most significant bit (MSB)
- ROR provides the value of the contents of a register rotated by a value

Algorithm:

- set the value of offset SI equal to 500.
- set the value of offset DI equal to 600.
- load the value 0000 into register AX.
- load the data of AX register into DS(data segment).

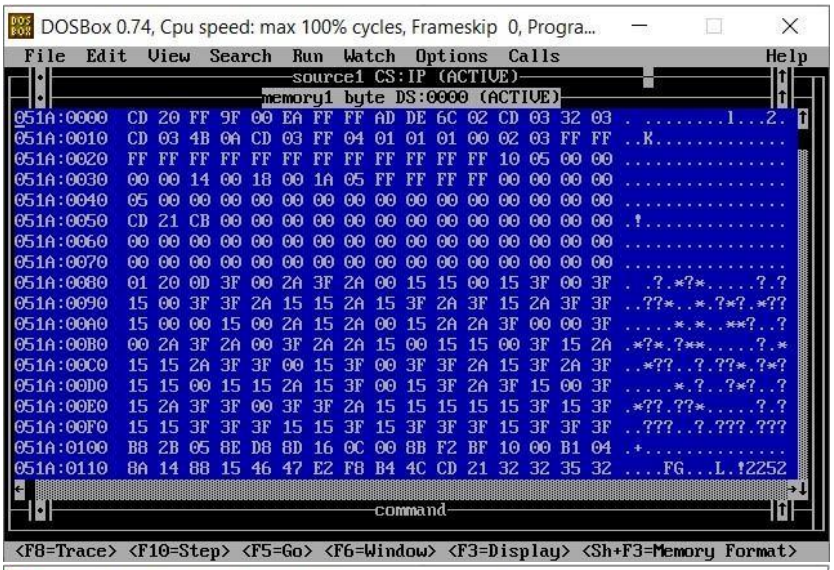
- load the data of AX register into ES(extra segment).
- load the data of offset SI into CL register and load value 00 into CH register.
- increment the value of SI by one.
- clear the directional flag so that data is read from lower memory to higher memory location.
- check the value of CX, if not equal to zero then repeat step 10 otherwise go to step 11.
- transfer the data from source memory location to destination memory location and decrease the value of CX by one.
- Stop

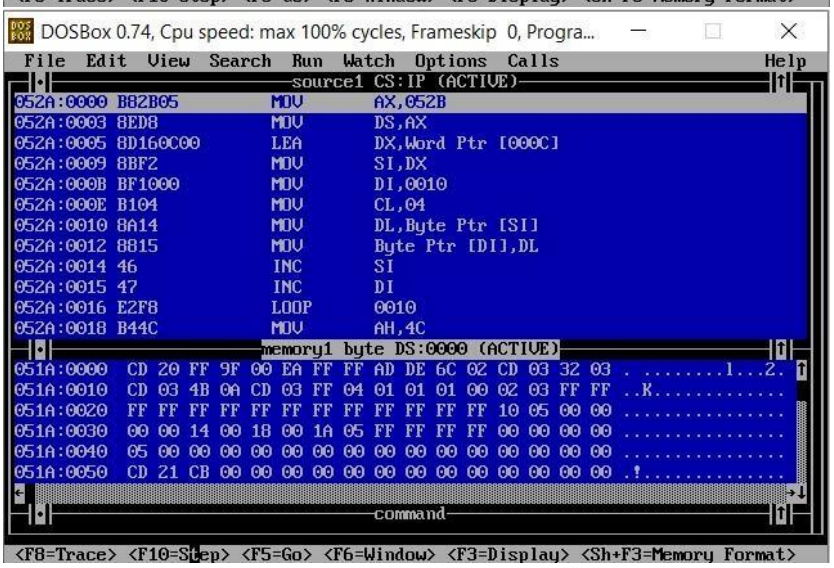
Code:

```
.model SMALL
.STACK 100H
.DATA
MSG DB '2252'

.CODE
MAIN PROC
MOV AX,@DATA
MOV DS,AX
LEA DX,MSG
MOV SI,DX
MOV DI,16 MOV
CL,04 LAB:
MOV DL,[SI]
MOV [DI],DL
INC SI
INC DI
LOOP LAB
MOV AH,4CH
INT 21H
MAIN ENDP
END MAIN
```

Department of Information Technology

Output:





Conclusion:

A block of memory can be transferred from one location to another location in MASM using very few instructions. This can be done using two different methods, one using string instructions and the other without using string instructions. Both these methods are documented above.

2. Program to count number of 1's and 0's in a given 8-bit number

Theory:

In this program we are using the rotate operation to count the number of 1's. As there are 8 different bits in 8-bit number, then we are rotating the number eight times. We can use RRC or RLC. Here we have used the RRC instruction. This instruction sends the LSb to MSb also to carry flag. So after each iteration we can check the carry status to get the count of 1s.

If the number is DA (11011010) then the answer will be 5, as there are five 1s in the number.

Instruction to be explain:

- MOV AX, @DATA – it is the first line of code that gets run. @DATA is a variable that holds the value of the location in memory where the data segment.
- MOV is used to transfer the data from memory to accumulator
- ADD is used to add accumulator with any of register
- DAA is used to check if sum > 9
- JNC is used jump if no carry to given memory location
- INR is used to increase given register by 1
- INT generates a software interrupt. It takes the interrupt number formatted as a

byte value.

- AND instruction is used for supporting logical expressions by performing bitwise AND operation
- ROL rotates the bits within the destination operand to the left, where left is toward the most significant bit (MSB)
- ROR provides the value of the contents of a register rotated by a value

Algorithm:

Step 1: Load register A (*accumulator*) with the given data
Step 2: Load register B with 08H to set up a decrement counter

Department of Information Technology

Step 3: Load register C as counter to count the numbers of 1's *(initial value 00H)*
Step 4: Load register D as counter to count the number of 0's *(initial value 00H)*
Step 5: Rotate the content of Accumulator to left through carry

Step 6: If on carry from *Step 5* then jump directly to *Step 9*
Step 7: Else increase counter C by 1

Step 8: Jump unconditionally to *Step 10*
Step 9: Increase counter D by 1
Step 10: Decrease counter B by 1

Step 11: Until B is not equal to 0 repeat from *Step 5*

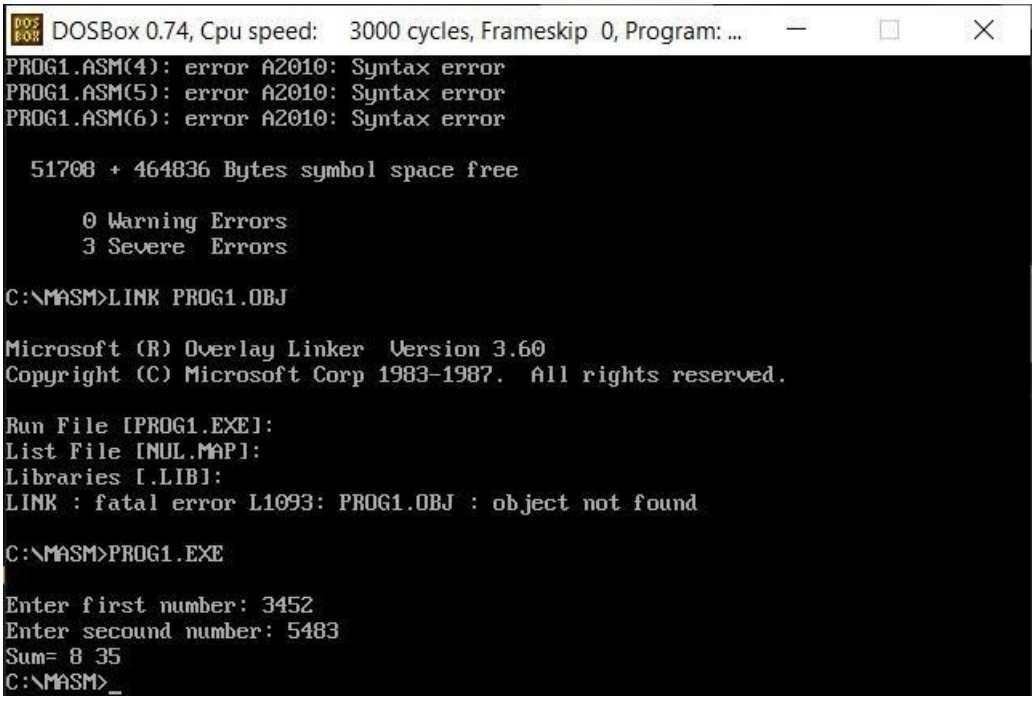
Code

```
.MODEL SMALL
.STACK 100H
.DATA
MSG DB 10,13 'Enter 8 Bit Number: $'
MSG2 DB 10,13 'Number of Zero: $'
MSG1 DB 10,13 'Number of One: $'

.CODE
MAIN PROC
MOV AX,@DATA
MOV DS,AX
LEA DX,MSG
MOV AH,9
INT 21h
MOV AH,01H
INT 21h sub
AL,30H
MOV CL,04H
```

```
SHL AL,CL
MOV BH,AL
MOV AH,01H
INT 21h
SUB AL,30H
ADD BH,AL
MOV AX,BX
MOV DX,00H
MOV CX,08H back:
ROL BX,01H
JNC next
INC DL next:
JC next2 INC
DH next2:
LOOP back
MOV BX,DX
ADD BL,30H
ADD BH,30H
LEA DX,MSG1
MOV AH,09H
INT 21H
MOV DL,BL
MOV AH,02H
INT 21H
LEA DX,MSG2
MOV AH,09H
INT 21H
MOV DL,BH
```

```
MOV AH,02H INT
21H
MOV AH,4CH
INT 21H
MAIN ENDP
END MAIN
```

Output:



Conclusion:

A binary number given in 8-bit representation can vary from 0 to 2^8 i.e. 0 to 255. This 8-bit representation can have a maximum of 8 ones or 8 zeroes or any combination of ones and zeroes between that. The number of ones and zeroes can be found out by using LOOP, JNC, ROL instructions in MASM.

3. Program to find even and odd numbers from a given list

Theory:

A number is said to be odd if its lower bit is 1 otherwise even. Therefore to identify whether the number is even or odd, we perform AND operation with 01 by the help of **ANI** instruction. If number is odd then we will get 01 otherwise 00 in accumulator. **ANI** instruction also affect the flags of 8085. Therefore if accumulator contains 00 then zero flag becomes set otherwise reset.

Instruction to be explain:

- MOV AX, @DATA – it is the first line of code that gets run. @DATA is a variable that holds the value of the location in memory where the data segment.
- MOV is used to transfer the data from memory to accumulator
- ADD is used to add accumulator with any of register
- DAA is used to check if sum > 9
- JNC is used jump if no carry to given memory location
- INR is used to increase given register by 1
- INT generates a software interrupt. It takes the interrupt number formatted as a

byte value.

- AND instruction is used for supporting logical expressions by performing bitwise AND operation
- ROL rotates the bits within the destination operand to the left, where left is toward the most significant bit (MSB)
- ROR provides the value of the contents of a register rotated by a value

Algorithm:

1. Load the content of memory location 2050 in accumulator A.
2. Perform AND operation with 01 in value of accumulator A by the help of **ANI** instruction.

3. Check if zero flag is set, i.e if ZF = 1 then store 22 in accumulator A otherwise store 11 in A.
4. Store the value of A in memory location 3050

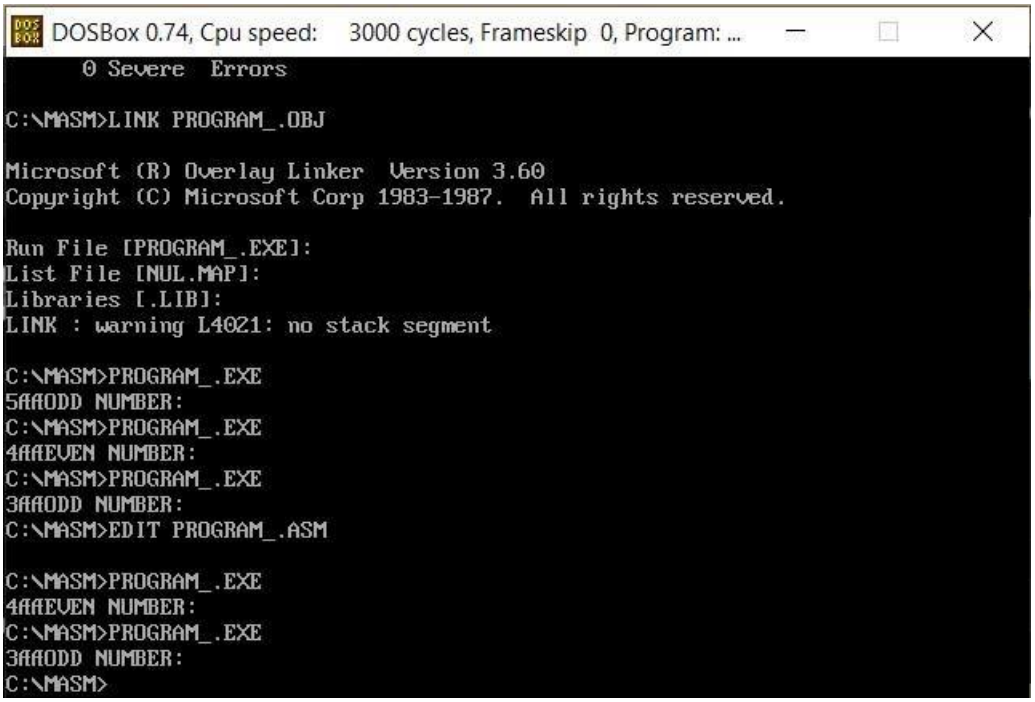
Code:

```
DATA SEGMENT
EV DB 'EVEN NUMBER: $'
OD DB 'ODD NUMBER: $'
DATA ENDS
CODE SEGMENT ASSUME
CS:CODE, DS:DATA BEGIN:
MOV AX,DATA
MOV DS,AX
MOV AH,1
INT 21H
MOV BL,2 DIV
BL
CMP AH,0
JE EVENNUMBER
MOV AH,10
MOV AH,2
INT 21H
MOV AH,13
MOV AH,2
INT 21H
MOV DX,OFFSET OD
MOV AH,9
INT 21H
MOV AH,4CH
INT 21H
```

Department of Information Technology

```
EVENNUMBER:
MOV AH,10
MOV AH,2
INT 21H
MOV AH,13
MOV AH,2
INT 21H
MOV DX,OFFSET EV
MOV AH,9
INT 21H
MOV AH,4CH
INT 21H
CODE ENDS
END BEGIN
```

Output:



Conclusion:

In this experiment we can see that the even and odd numbers were identified by performing AND operation with 01 by the help of ANI instruction such that If number is odd then we will get 01 otherwise 00 in accumulator.

4. Program to search for a given number

Theory:

In this program we are taking only 5 numbers. We are searching the number 25. after successful search the DX register will hold the offset address, and BX register will hold the index of that number. We are taking each number from that array and then compare it with 25. If the numbers are same, then we will return the address and index. **Instruction to be explain:**

- MOV AX, @DATA – it is the first line of code that gets run. @DATA is a variable that holds the value of the location in memory where the data segment.
- MOV is used to transfer the data from memory to accumulator
- ADD is used to add accumulator with any of register
- DAA is used to check if sum > 9
- JNC is used jump if no carry to given memory location
- INR is used to increase given register by 1
- INT generates a software interrupt. It takes the interrupt number formatted as a

byte value.

- AND instruction is used for supporting logical expressions by performing bitwise AND operation
- ROL rotates the bits within the destination operand to the left, where left is toward the most significant bit (MSB)
- ROR provides the value of the contents of a register rotated by a value

Algorithm:

Department of Information Technology

1. Move 2000 in AX and assign it to ES
2. Assign value 600 to DI
3. Move 25 in AL
4. Move 0005 in CX
5. Move the contents of CX to BX
6. Clear the value of Directional Flag (DF)
7. Repeat step 7 till Zero Flag (ZF) is not set
8. Scan byte from [DI] and check its difference with contents of AL. Update the value of DI
9. Decrements the value of DI by 1
10. Subtract the value of BX by CX
11. Decrements the value of BX by 1
12. Halt the program

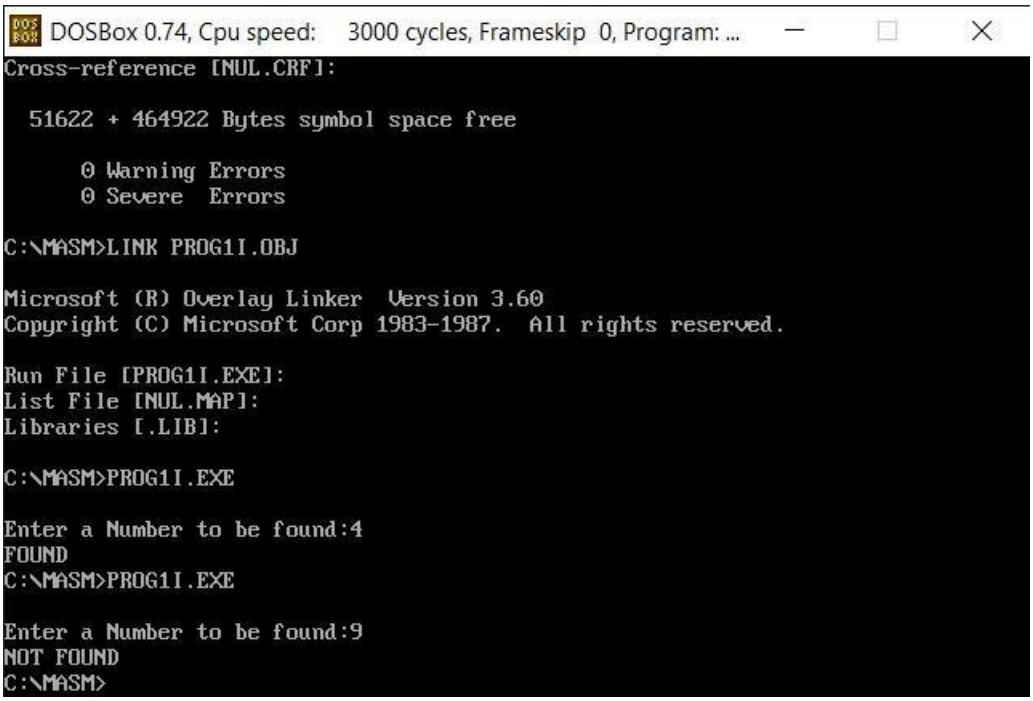
Code:

```
.MODEL SMALL

.STACK 100H

.DATA

ARR DB '45637'

MSG1 DB 10,13, 'Enter a Number to be found:$'

MSG2 DB 10,13, 'FOUND$'

MSG3 DB 10,13, 'NOT FOUND$'

.CODE

START:

MOV AX,@DATA

MOV DS,AX

LEA DX,MSG1

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

MOV BH,AL

LEA DX,ARR
```

```
MOV SI,DX
MOV CL,05 LAB:
MOV DL,[SI]
CMP DL,BH
JZ FOUNDNO
INC SI
LOOP LAB
LEA DX,MSG3
MOV AH,9

INT  21H  JMP
ENDPRG
FOUNDNO:
LEA DX,MSG2
MOV AH,9
INT 21H
ENDPRG:
MOV AH,4CH
INT 21H
END START
```

Output:

Department of Information Technology



<u>Conclusion</u>:

When an array of numbers is given as input, a specific number can be searched in that array with the help of a few instructions like JZ, LOOP, JMP, and INC.